Lineair Programmeren met Excel

Stijn Lievens

Hogeschool Gent
2018–2019

Doel van het practicum

Het doel van het practicum is drieledig: enerzijds is het de bedoeling om Excel te leren gebruiken om (I)LP-problemen op te lossen, anderzijds is er ook nog de mogelijkheid om nog eens te oefenen met het *opstellen* van een wiskundig model. Tenslotte zien we hoe een graafprobleem kan geformuleerd worden als een (I)LP probleem.

Oplossen van een LP-probleem

Een eenvoudig probleem met twee beslissingsvariabelen

We gebruiken de Excel-solver om een LP-probleem op te lossen.

We beschouwen volgend LP-probleem uit de cursus (Voorbeeld 7.32):

$$\max D(x_1, x_2) = 3x_1 + 2x_2$$

onder de beperkingen

$$\begin{cases} -4x_1 + 6x_2 \leq 21 \\ 2x_1 + 6x_2 \leq 39 \\ 4x_1 + 2x_2 \leq 33. \end{cases}$$

De niet-negativiteitsvoorwaarden zijn van kracht. Gebruik de Excel-solver om dit probleem op te lossen[1].

Een iets groter probleem en een extra moeilijkheid

Los volgend LP-probleem op:

$$\max D(x_1, x_2, x_3, x_4) = -2x_1 + x_2 - 4x_3 + 3x_4$$

[1] Op YouTube zijn er heel wat video's die tonen hoe je dit moet doen, bv. `https://www.youtube.com/watch?v=RicajFzoenk`

onder de beperkingen:

$$\begin{cases} x_1 + x_2 + 3x_3 + 2x_4 \leq 4 \\ x_1 - x_3 + x_4 \geq 1 \\ 2x_1 + x_2 \leq 2 \\ x_1 + 2x_2 + x_3 + 2x_4 = 2 \end{cases}$$

De niet-negativiteitsvoorwaarden zijn van kracht. Gebruik Excel of R om de oplossing te vinden.

Veronderstel nu dat het niet langer nodig is dat $x_1 \geq 0$, m.a.w. x_1 mag nu een willekeurig reëel getal zijn. Hoe kan je het probleem toch formuleren als een LP-probleem? Merk op dat x_1 kan geschreven worden als

$$x_1 = x_1^+ - x_1^-,$$

waarbij zowel x_1^+ als x_1^- niet-negatieve getallen zijn. Los opnieuw op.

Geheeltallig lineair programmeren

Knapzakprobleem

Los het knapzakprobleem op waarvoor de gewichten en de waarden van de items die kunnen meegenomen worden gegeven zijn in het bestand `knapzak.xlsx`. De rugzak kan een maximum gewicht van 400 decagram dragen. Deze waarden en gewichten zijn overgenomen van het voorbeeld op deze URL: `https://www.rosettacode.org/wiki/Knapsack_problem/0-1`.

Opstellen van een wiskundig model voor een toewijzingsprobleem

Los oefening 3 van sectie 8.6 uit de cursus op.

Extra: Kortste pad problemen

Het algoritme van Dijkstra uit de cursus (Algoritme 5.6) geeft een oplossing voor het vinden van het kortste pad van één knoop naar alle andere knopen uit de graaf. Het algoritme is eenvoudig aan te passen om slechts het kortste pad te vinden tussen een startknoop en één specifieke doelknoop: het algoritme kan gestopt worden op het moment dat de doelknoop van de prioriteitswachtrij Q wordt *verwijderd*.

We tonen nu aan hoe we dit probleem (van het vinden van het kortste pad van een startknoop naar een bepaalde doelknoop) kunnen formuleren als een (I)LP probleem dat vervolgens kan opgelost worden m.b.v. de Excel solver.

Veronderstel dat de graaf $G = (V, E)$ knopen 1 t.e.m. n heeft en dat we het kortste pad wensen te vinden van knoop s naar knoop t. Veronderstel verder dat het gewicht van de boog (i, j) gegeven wordt door $w_{i,j}$.

Voor elke boog $(i, j) \in E$ moeten we weten of die boog deel uitmaakt van het korste pad tussen s en t of niet. Hiervoor gebruiken we de beslissingsvariabelen $x_{i,j}$:

$$x_{i,j} = \begin{cases} 1 & \text{als boog } (i,j) \text{ deel uitmaakt van het (gekozen) kortste pad } \textit{in die richting} \\ 0 & \text{anders.} \end{cases}$$

We wensen het totale gewicht van het gekozen pad te minimaliseren. De te minimaliseren doelfunctie is m.a.w.

$$\sum_{(i,j)\in E} w_{i,j} x_{i,j}.$$

Hoe kunnen we nu uitdrukken dat de gekozen bogen een pad vormen van s naar t? In de startknoop s moet er juist één boog vertrekken en er mag geen enkele boog toekomen[2]:

$$\sum_{i \,|\, (s,i)\in E} x_{s,i} - \sum_{i \,|\, (i,s)\in E} x_{i,s} = 1.$$

In de doelknoop t moet er steeds één boog toekomen en er mag geen enkele boog vertrekken:

$$\sum_{i \,|\, (t,i)\in E} x_{t,i} - \sum_{i \,|\, (i,t)\in E} x_{i,t} = -1.$$

Wat met de andere knopen? Op een willekeurige knoop j op het kortste pad van s naar t geldt dat er juist één boog toekomt en juist één boog vertrekt; het aantal vertrekkende bogen is gelijk aan het aantal toekomende bogen. Dit is eveneens waar voor een knoop j *niet* op het kortste pad van s naar t (want daar is het aantal toekomende en vertrekkende bogen gelijk aan nul):

$$\underbrace{\sum_{i \,|\, (j,i)\in E} x_{j,i}}_{\text{bogen die vertrekken in } j} - \underbrace{\sum_{i \,|\, (i,j)\in E} x_{i,j}}_{\text{bogen die toekomen in } j} = 0.$$

Samenvattend moeten we om het kortste pad te vinden van s naar t het volgende (I)LP oplossen:

$$\min \sum_{(i,j)\in E} w_{i,j} x_{i,j}.$$

onder de beperkingen:

$$\sum_{i \,|\, (j,i)\in E} x_{j,i} - \sum_{i \,|\, (i,j)\in E} x_{i,j} = \begin{cases} 1 & \text{als } j = s \\ -1 & \text{als } j = t \\ 0 & \text{anders} \end{cases} \quad \text{voor } j \in \{1, 2, \ldots, n\}$$

[2]Strikt genomen drukt de onderstaande formule een iets minder restrictieve eigenschap uit, namelijk dat het aantal vertrekkende bogen precies één meer is dan het aantal inkomende bogen. Dit blijkt echter voldoende te zijn.

Opmerking: normaliter zou je hier ook de beperkingen $x_{i,j} \in \{0,1\}$ verwachten. Echter: door de *bijzondere vorm* van dit LP probleem is dit niet nodig. De optimale (basis)oplossing zal steeds de eigenschap hebben dat een boog ofwel volledig wel ofwel volledig niet wordt gekozen. Wanneer we het "gewone" LP-probleem oplossen zullen we in dit geval (door de bijzondere vorm van het probleem) gegarandeerd een oplossing bekomen waarvoor alle beslissingsvariabelen binair zijn.

Gebruik bovenstaande methode om het kortste pad te vinden tussen Brugge en Aarlen in de graaf van Figuur 5.4 uit de cursus. Je kan hiervoor starten met het Excelwerkblad `kortstepad.xlsx`. De kolommen `B` (**Van**) en `C` (**Naar**) bevatten de namen van de steden. De afstand tussen de steden wordt gegeven in kolom `D` (**Afstand**). De beslissingsvariabelen komen in kolom `E` (**Op Pad?**) en het is de bedoeling dat de cel `F54` de totale afstand bevat van het gevonden pad.

In de kolom `J` (**Flow Uit**) wordt

$$\underbrace{\sum_{i \mid (j,i) \in E} x_{j,i}}_{\text{bogen die vertrekken in } j}$$

berekend voor de stad j die aangegeven staat in de overeenkomstige rij in kolom `I` (**Stad**). Om deze uitdrukking te berekenen kan je handig gebruikmaken van de Excelfunctie

```
SOM.ALS(bereik; criterium; optelbereik)
```

In dit geval willen we alle beslissingsvariabelen (`optelbereik`) optellen waarvoor de stad in de **Van**-kolom (`bereik`) gelijk is aan de stad in de overeenkomstige rij van kolom `I`. Dit laatste is uiteraard het `criterium` dat moet worden gebruikt voor `SOM.ALS`.

Op een gelijkaardige manier berekent men de correcte waardes voor de **Flow In** kolom. Hieruit volgen dan onmiddellijk de waarden voor de **Netto Flow** kolom.

Het is nu eenvoudig om met de Solver de optimale oplossing te vinden en hieruit het corresponderende kortste pad te bepalen.

Geef nu het kortste pad van Brugge naar Aarlen.